

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-224155

(P2000-224155A)

(43) 公開日 平成12年8月11日 (2000.8.11)

(51) Int.Cl.⁷

識別記号

F I

テーマコード(参考)

H 0 4 L 9/08

H 0 4 L 9/00

6 0 1 B 5 J 1 0 4

9/16

6 0 1 E 5 K 0 3 0

12/18

6 4 3 9 A 0 0 1

11/18

審査請求 未請求 請求項の数 4 O L (全 10 頁)

(21) 出願番号

特願平11-20190

(22) 出願日

平成11年1月28日 (1999.1.28)

(71) 出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72) 発明者 塩野崎 敦

東京都品川区東五反田3丁目14番13号 ソ

ニーコンピュータサイエンス研究所内

(72) 発明者 濱野 淳史

神奈川県横浜市港北区日吉3-14-1 慶

應義塾大学大学院内

(74) 代理人 100067736

弁理士 小池 晃 (外2名)

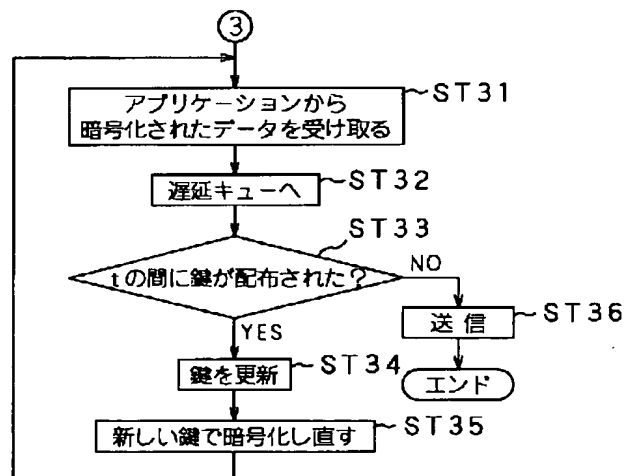
最終頁に続く

(54) 【発明の名称】 ネットワークシステム及びデータ送受信方法

(57) 【要約】

【課題】 鍵の配布や鍵を用いたデータの送受信の順番に関係なく、リアルタイム性を維持してマルチキャスト通信を行う。

【解決手段】 遅延キューにデータが供給されてから出力するまでの時間 t の間に共有鍵が配布されたかを判定する (ステップ S T 3 3)。共有鍵が配布されたと判定したときは共有鍵を更新し (ステップ S T 3 4)、データの暗号化をし直す (ステップ S T 3 5)。また、共有鍵が配布されていないと判定したときはそのデータをそのまま送信する (ステップ S T 3 6)。



【特許請求の範囲】

【請求項 1】 鍵サーバと複数のクライアントからなり、上記鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおいて、上記各クライアントは、

上記鍵サーバに問い合わせをして当該鍵サーバから上記各クライアントに上記共有鍵が送信されるまでの保証されている遅延時間を算出する算出手段と、送信対象となる暗号化済みのデータに対して、上記算出手段により算出された遅延時間に相当する時間の遅延処理を施す遅延手段と、

上記遅延手段の遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを上記遅延手段に供給し、上記遅延手段の遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延手段から出力される暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信する送信手段とを備えることを特徴とするネットワークシステム。

【請求項 2】 上記各クライアントは、上記送信手段からの暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信する第 1 の動作モードと、

上記送信対象となる暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信すると共に、当該暗号化済みのデータを最も遠いクライアントまで送信する時間に相当する時間記憶し、上記暗号化済みのデータの再送要求があったときは再送要求のあったクライアントに対して当該暗号化済みのデータを再送する第 2 の動作モードとを切換制御することを特徴とする請求項 1 記載のネットワークシステム。

【請求項 3】 鍵サーバと複数のクライアントからなり、上記鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおけるデータ送受信方法において、

上記各クライアントは、上記鍵サーバに問い合わせをして当該鍵サーバから上記各クライアントに上記共有鍵が送信されるまでの保証されている遅延時間を算出し、送信対象となる暗号化済みのデータに対して、上記算出

された遅延時間に相当する時間の遅延処理を施し、上記遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを再び上記遅延処理の対象とし、上記遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延処理の施された暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信することを特徴とするデータ送受信方法。

【請求項 4】 上記各クライアントは、

10 上記遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを上記遅延処理対象とし、上記遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延処理の施された暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信する第 1 の動作モードと、

上記送信対象となる暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信すると共に、当該暗号化済みのデータを最も遠いクライアントまで送信する時間に相当する時間記憶し、上記暗号化済みのデータの再送要求があったときは再送要求のあったクライアントに対して当該暗号化済みのデータを再送する第 2 の動作モードとを切換制御することを特徴とする請求項 3 記載のデータ送受信方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、1つの共通鍵を用いてマルチキャスト通信を行うネットワークシステム及びデータ送受信方法に関する。

30 【0002】

【従来の技術】インターネット等の広域分散ネットワークにおいてマルチキャスト技術は必要不可欠である。現在でも m b o n e を始めとする仮想ネットワークでマルチキャスト通信は実現されているが、実際にエンドユーザに提供できる出力の品質はかなり低い。現在、ネットワークの高速化及び新技術の開発などに伴い、品質向上、さらに得ることのできるサービス品質の制御、Quality of Service (Q o S) 技術の研究が盛んに行われている。そこで、今後マルチキャストアプリケーションはさらにリアルタイム性を向上することが求められている。

【0003】鍵交換機構を必要とするマルチキャストアプリケーションは、通常、鍵を配布する鍵サーバとグループのメンバによって構成される。鍵サーバは、グループの一員であってもかまわない。通常、鍵はこのグループ単位で共有される。

【0004】マルチキャストアプリケーションの場合、グループに参加しているメンバの構成は動的に変化する。すなわち、任意の時間において新たにメンバが参加したり、既存のメンバが脱退することが可能である。こ

3

のような場合、マルチキャスト通信によるデータが盗聴されることを防止するために新しい鍵を生成し、現存するメンバにその鍵を配布する必要がある。

【0005】鍵交換機構を導入したマルチアプリケーションの場合、鍵の更新を行うための特別な処理が必要である。しかし、全てのメンバが同じ鍵を得るためには、一旦、アプリケーションの動作を止めることとなる同期をとり、鍵の一貫性を保証するための処理が必要であった。

【0006】しかし、リアルタイムで動作するマルチキャストアプリケーションの場合、実行が中断されることは回避しなければならない。

【0007】

【発明が解決しようとする課題】アプリケーションの実行を中断しない解決策としては、因果順序(Causal Ordering)を用いたものが提案されている。なお、以下の例では、鍵サーバとクライアントたるメンバA及びメンバBによりネットワークが構成されているものとする。

【0008】図9に示すように、鍵サーバは、メンバの構成が変わると、新しい鍵K'を古い鍵Kで暗号化して({K'}K)、メンバA及びメンバBに配布する。メンバBは、新しい鍵K'を用いて送信すべきメッセージMを暗号化して({M}K')、鍵サーバ及びメンバAに送信している。ここで、メンバAは、ネットワークの遅延によって、({M}K')を受け取ってから、新しい鍵K'を鍵サーバから受け取っている。メンバAは、通常ではメンバBからのメッセージMを廃棄してしまうおそれがあるが、メンバBにおいて付加された因果情報を参照することによって無事メッセージMを新しい鍵で復号することができる。

【0009】つぎに、図10に示すように、メンバBがメッセージMについて鍵Kを用いて暗号化し({M}K)、これを鍵サーバ及びメンバAに送信した場合について説明する。このとき、鍵サーバは、メンバBから({M}K)を受信してから、新たな鍵K'を古い鍵Kで暗号化して({K'}K)、メンバA及びメンバBに送信している。メンバAは、鍵サーバから({K'}K)を受け取ってから、メンバBからの({M}K)を受信している。メンバAは、因果順序による順序づけを行うことによって、メンバBから送られるメッセージ({M}K)を正しく復号することができる。すなわち、アプリケーションを中断し、最新鍵の同期をとる必要はない。

【0010】しかし、図11の場合は、問題が発生する。すなわち、鍵サーバが({K'}K)をメンバA及びメンバBに配布し、その直後にメンバBが({M}K)を鍵サーバ及びメンバAに送信し、メンバAが({K'}K)の後に({M}K)を受信した場合である。

【0011】このとき、鍵サーバは新しい鍵K'を送信し、さらにメンバBは古い鍵Kで暗号化したメッセージMを送信する。しかし、この2つの転送に対しては因果

4

情報を付加することができない。そこで、メンバAは、復号できない({M})に対する再送要求をメンバBに発行し、新たな鍵K'で暗号化されたメッセージMを送信するように要求する。

【0012】このような再送要求機構を用いることによって、一時的には上述した問題を解決することができる。しかし、実際には、メンバBは、再送要求に備えてメッセージMを保持しておく必要がある。メンバAがどの程度再送要求が発生するかを予測することはできないので、メンバBが保持しておくメッセージMの数を決定することも困難である。

【0013】ネットワーク遅延を決定することできない場合は、理論的には無限のメッセージを保持しなければならない。この場合、メンバAは、({M})Kの再送要求を行ったことによって、再度メッセージMを受け取るまでの予測不可能な遅延が発生する。これにより、リアルタイム性を維持することができなくなり、また、連続的にアプリケーションにデータを供給することができなくなる。

【0014】本発明は、このような実情に鑑みて提案されたものであり、鍵の配布や鍵を用いたデータの送受信の順番に関係なく、リアルタイム性を維持してマルチキャスト通信を行うことができるネットワークシステム及びデータ送受信方法を提供することを目的とする。

【0015】

【課題を解決するための手段】上述の課題を解決するために、本発明に係るネットワークシステムは、鍵サーバと複数のクライアントからなり、鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおいて、各クライアントは、鍵サーバに問い合わせをして当該鍵サーバから各クライアントに共有鍵が送信されるまでの保証されている遅延時間を算出する算出手段と、送信対象となる暗号化済みのデータに対して、算出手段により算出された遅延時間に相当する時間の遅延処理を施す遅延手段と、遅延手段の遅延処理中に鍵サーバから新たな鍵が配布されたときは、新たな鍵によってデータを再暗号化して当該暗号化されたデータを遅延手段に供給し、遅延手段の遅延処理中に鍵サーバから新たな鍵が配布されなかったときは、遅延手段から出力される暗号化済みのデータを他のクライアント及び鍵サーバに送信する送信手段とを備えることを特徴とする。

【0016】本発明に係るデータ送受信方法は、鍵サーバと複数のクライアントからなり、上記鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各ク

クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおけるデータ送受信方法において、上記各クライアントは、上記鍵サーバに問い合わせをして当該鍵サーバから上記各クライアントに上記共有鍵が送信されるまでの保証されている遅延時間を算出し、送信対象となる暗号化済みのデータに対して、上記算出された遅延時間に相当する時間の遅延処理を施し、上記遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを再び上記遅延処理の対象とし、上記遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延処理の施された暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信することを特徴とする。

【0017】

【発明の実施の形態】以下、本発明の実施の形態について、図面を参照しながら説明する。

【0018】本発明は、例えば図1に示す構成のネットワークシステム1に適用することができる。

【0019】上記ネットワークシステム1は、ネットワーク間で1つの鍵を共有するマルチキャスト通信を行うものであり、クライアントたるメンバ2(2A, 2B, 2C, ...)と暗号化/復号のための鍵を配布する鍵サーバ3とがネットワーク4を介して接続されている。

【0020】マルチキャスト通信は、第三者の盗聴を防止するために、新たなメンバが参加したりメンバが脱退したときには、共有している鍵(以下、「共有鍵」という。)の更新を行う。このような共有鍵の更新は、鍵サーバ3が新たな共有鍵を生成して各メンバに配布することによって行われる。なお、鍵サーバ3によって配布される共有鍵が全てのメンバ2に送信されることを保証するために、鍵配布のためのマルチキャスト通信は信頼性のあるものを用いる。さらに、鍵サーバ3が各メンバ2間において新しい共有鍵を配布するために必要な時間 t は、予約資源プロトコル等によって保証されているものとする。

【0021】メンバ2は、図2に示すように、共有鍵を用いた暗号化/復号等の所定のデータ処理等を行うアプリケーション21と、因果順序マルチキャスト機構22とを備える。因果順序マルチキャスト機構22は、暗号化済みのデータを送信する送信モジュール23と、暗号化されたデータや共有鍵を受信する受信モジュール24とを備える。

【0022】送信モジュール23は、暗号化されたデータに遅延を与える遅延キュー25と、遅延キュー25からのデータに因果順序を示すデータであるベクタタイム

スタンプを挿入するベクタタイムスタンプ機構26とを有する。

【0023】遅延キュー25は、アプリケーション21からのデータに対して、常に一定時間の遅延を与えてから出力するように制御されている。ここで、一定時間とは、鍵サーバ3が各メンバ2に共有鍵を配布するまでの時間 t をいう。

【0024】ベクタタイムスタンプ機構26は、ベクタタイムスタンプを生成し、これを遅延キュー25からのデータに挿入して、他のメンバ2及び鍵サーバ3に送信する。

【0025】以下、プロセス(メンバ) p_i のベクタタイムスタンプを $VT(p_i)$ と表記し、プロセス p_i のベクトル要素を $VT(p_i)[i]$ で表す。また、送信されるデータに挿入するベクタタイムスタンプを $VT(m)$ と表記する。なお、ここで用いるベクタタイムスタンプは[1]に基づくものとする。

【0026】ベクタタイムスタンプの更新アルゴリズム以下に示す。

1. プロセス起動時に $VT(p_i)$ の全要素を0に初期化する。

【0027】2. 送信時に $VT(p_i)[i]$ を1増加する。

【0028】3. 送信データに $VT(p_i)$ を挿入する(以下、これを「 $VT(m)$ 」とする)。

【0029】4. p_i からのデータを p_j が受信した場合には(1)式に従って、 $VT(p_j)$ を更新する。

【0030】

【数1】

$$VT(p_j)[k] = \max(VT(p_j)[k], VT(m)[k]) \quad (1)$$

ただし、 n をメンバ数とすると、 $\{vk : k \in n\}$ である。

【0031】一方、受信モジュール24は、受信したデータに挿入されているベクタタイムスタンプを検査するベクタタイムスタンプ機構27と、アプリケーション21へのデータの配送を一時抑止する抑止キュー28と、抑止キュー28からのデータを鍵サーバアプリケーション21に配送する配送キュー29とを備える。

【0032】ベクタタイムスタンプ機構27は、データを受信したときに、当該データに挿入されているベクタタイムスタンプが(2)式の条件を満たしているかを判定する。

【0033】

【数2】

$$\begin{cases} VT(m)[k] = VT(p_j)[k] + 1 & k = i \\ VT(m)[k] \leq VT(p_j)[k] & k \neq i \end{cases} \quad (2)$$

ただし、 n をメンバ数とすると、 $\{vk : k \in n\}$ である。

【0034】ベクタタイムスタンプ機構 27 は、(2) 式の条件を満たしていない場合には、受信したデータを抑止キュー 28 に供給する。また、ベクタタイムスタンプ機構 27 は、(2) 式の条件を満たしている場合は、
10 配送キュー 29 を介してアプリケーション 21 に供給し、ベクタタイムスタンプを(1)式に従って更新する。

【0035】つぎに、このようなベクタタイムスタンプの更新アルゴリズムについて、簡単な具体例を挙げながら説明する。図 3 に示すように、メンバが A、B、C からなるグループがあるものとする。A、B、C は、それぞれ起動時には、ベクタタイムスタンプの全要素が 0 である (0, 0, 0) を持つ。

【0036】A は、B 及び C に対してデータを送信するとき、(1, 0, 0) を VT (m) として上記データに挿入し、当該データを B 及び C に送信する。このとき、自己の VT (A) も (1, 0, 0) に更新する。
20

【0037】C は、ベクタタイムスタンプ機構 27 で A からのデータを受信すると、自己の VT (C) と VT (m) とを比較する。このとき (2) 式を満足するので、ベクタタイムスタンプ機構 27 は、データを配送キュー 29 に供給し、VT (C) を (1, 0, 0) に更新する。その後、C がデータを送信する場合には、VT (m) を (1, 0, 1) としてデータに挿入し、A 及び
30 B に送信する。

【0038】ここで、B は、C からのデータを受信した後、A からのデータを受信する。B は、C からのデータをベクタタイムスタンプ機構 27 で受信し、そのベクタタイムスタンプが (2) 式を満たしているか判定する。ここでは、(2) 式を満たさないで、ベクタタイムスタンプ機構 27 は、上記データを抑止キュー 28 に供給する。

【0039】B は、A からのデータを受信する。B のベクタタイムスタンプ機構 27 は、VT (m) が (1, 0, 0) である A からのデータを受信すると (2) 式を満足するので、A からのデータを配送キュー 29 に供給する。このとき、VT (B) は (1, 0, 0) に更新され、抑止キュー 28 に抑止されていたデータは、(2) 式を満足するようになる。そして、抑止キュー 28 は、抑止していた C からのデータを配送キュー 29 を介してアプリケーション 21 に供給する。

【0040】これにより、B は、A からのデータが C からのデータに遅れて到着した場合であっても、ベクタタイムスタンプを用いて因果順序を満足させて、A からの
50

データ、C からのデータの順にアプリケーション処理を行うことができる。

【0041】なお、ベクタタイムスタンプの要素数は、グループのメンバシップが変わる度に変化する。具体的には、1 のメンバが新たに参加するとベクタタイムスタンプの要素は 1 つ大きくなり、1 のメンバが脱退するとその要素は 1 つ小さくなる。

【0042】さらに、メンバ 2 のシステムとして、リライアブルマルチキャスト・資源予約プロトコル機構 51 と、IP マルチキャスト機構 52 とを備えている。

【0043】リライアブルマルチキャスト・資源予約プロトコル機構 51 は、マルチキャスト通信に信頼性を保証するものである。ここにいう信頼性とは、第 1 に、共有鍵が全てのメンバ 2 に送信されること、第 2 に、鍵サーバ 3 が各メンバ 2 間において新しい共有鍵を配布するために必要な時間が予約資源等によって保証されていることである。

【0044】IP マルチキャスト機構 52 は、マルチキャスト通信のインターネットプロトコルであり、上記リライアブルマルチキャスト・資源予約プロトコル機構 51 からのデータを他のメンバ 2 や鍵サーバ 3 に転送したり、また、受信したデータや共有鍵をリライアブルマルチキャスト・資源予約プロトコル機構 51 に供給する。

【0045】鍵サーバ 3 は、メンバ 2 とほぼ同様の構成となっている。すなわち、鍵サーバ 3 は、鍵サーバアプリケーション 31 と、因果順序マルチキャスト機構 32 とを備える。

【0046】鍵サーバアプリケーション 31 は、グループに参加しようとするユーザから参加メッセージを受信したりメンバが脱退したときのメッセージを受信したりすると、新たな共有鍵を生成し、この共有鍵を送信モジュール 33 に供給する。

【0047】送信モジュール 33 は、遅延キュー 35 と、ベクタタイムスタンプ機構 36 とを備え、送信モジュール 23 と同様の構成である。

【0048】遅延キュー 35 は、システムの実装上設けであるだけであり、共有鍵の配布時に当該共有鍵に遅延を与えるものではない。すなわち、鍵サーバアプリケーション 31 からの共有鍵は、すぐにベクタタイムスタンプ機構 36 に供給される。

【0049】ベクタタイムスタンプ機構 36 は、鍵サーバアプリケーション 31 からの共有鍵にベクタタイムスタンプを挿入して、各メンバ 2 に送信する。

【0050】受信モジュール 34 は、暗号化されたデータに挿入されたベクタタイムスタンプを検査するベクタタイムスタンプ機構 37 と、上記データを上述した (2) 式の条件を満たすまで抑止する抑止キュー 38 と、抑止キュー 38 から供給されるデータを鍵サーバアプリケーション 31 に配送する配送キュー 39 とを備
50

え、ベクタタイムスタンプ機構27と同様の構成となっている。

【0051】また、鍵サーバ3のシステムは、リライアブルマルチキャスト・資源予約プロトコル機構51及びIPマルチキャスト機構52を備えている。

【0052】すなわち、鍵サーバ3も、ネットワークシステム1を構成するメンバの一員であり、他のメンバ2と同様に、共有鍵にベクタタイムスタンプを挿入して各メンバに配布したり、共有鍵によって暗号化されたデータを受信している。

【0053】このようなネットワークシステム1は、第三者に盗聴されることなく安全にリアルタイムマルチキャスト通信を行うことができるように、このグループを構成するメンバの変化に応じた処理を行っている。

【0054】図4に示すように、ユーザがメンバとして上記ネットワークシステム1のグループに参加すると（ステップST1）、上記メンバ2は、鍵サーバ3とやり取りを行う（ステップST2）。

【0055】例えば、メンバとして参加するときは、鍵サーバ3に対して参加メッセージを送信する。鍵サーバ3は、参加メッセージを受信すると、他のメンバに対して新たな共有鍵を送信し、上記参加メッセージを送信したのに対して鍵の配布を行う。鍵が配布されることによって、メンバとして登録されたことになる。

【0056】新たに参加したメンバ2は、鍵サーバ3に対して問い合わせを行う。具体的には、メンバ2の因果順序マルチキャスト機構22が、鍵サーバ3の因果順序マルチキャスト機構32に対して、鍵サーバ3から各メンバ2において新しい共有鍵を配布するための必要な時間の問い合わせを行い、共有鍵を配布するための必要な時間 t を算出して、時間 t を決定する（ステップST3）。なお、因果順序マルチキャスト機構22は、上記時間 t を算出することができるのであれば、例えば鍵サーバ3のシステムであるリライアブルマルチキャスト・資源予約プロトコル機構51やその他の機構に直接問い合わせても良い。

【0057】その後、メンバ2がグループから脱退するときは図5に示すステップST11の処理に移行し、メンバ2がデータを受信するときは図6に示すステップST21に移行し、メンバ2がデータを送信するときは図7に示すステップST31に処理に移行する。

【0058】メンバ2は、図5に示すように、当該グループから脱退するときは（ステップST11）、鍵サーバ3に対して脱退メッセージを通知する（ステップST12）。鍵サーバ3は、上記脱退メッセージを受信すると、脱退するメンバ2を除いた各メンバ2に対して新たな鍵を配布する。これにより、メンバ2は、完全に脱退したことになる。

【0059】一方、メンバ2は、データを受信するときは、図6に示すステップST21以下の処理を実行す

る。

【0060】ステップST21において、メンバ2のベクタタイムスタンプ機構27は、他のメンバ2からのデータを受信すると、ステップST22に進む。

【0061】ステップST22において、ベクタタイムスタンプ機構27は、受信したデータに挿入されているベクタタイムスタンプの内容を検査し、因果関係があるかを判定する。ベクタタイムスタンプ機構27は、因果関係がない、すなわち受信したデータは因果順序に従っていないと判定したときはステップST23に進み、因果関係がある、すなわち受信したデータは因果関係に従っていると判定したときはステップST24に進む。

【0062】ステップST23において、ベクタタイムスタンプ機構27は、受信したデータを抑止キュー28に供給して、ステップST21に戻る。すなわち、次々に受信するデータが因果順序に従っていないときは、ステップST21からステップST23の処理が繰り返され、次々にデータが抑止キュー28に抑止されることになる。

【0063】ステップST24において、メンバ2のシステムのVT(p)を更新して、ステップST25に進む。

【0064】ステップST25において、ベクタタイムスタンプ機構27は、受信したデータを配送キュー29を介してアプリケーション21に供給し、ステップST26に進む。これにより、アプリケーション21は、自身が保有する共有鍵を用いてデータの復号を行い、さらに、所定の処理を実行する。

【0065】ステップST26において、抑止キュー28に抑止されているデータがあるかを判定する。抑止キュー28にデータがあるときはステップST27に進み、抑止キュー28にデータがないときは、メンバ2におけるデータ受信処理を終了する。

【0066】ステップST27において、抑止キュー28に抑止されているデータの因果関係を確認し、因果関係がある、すなわち抑止キュー28に抑止されているデータが因果順序に従っていると判定したときはステップST28に進む。また、因果関係がない、すなわち上記データは因果順序に従っていないと判定したときはステップST30に進む。

【0067】ステップST28において、メンバ2のシステムのVT(p)を更新して、ステップST28に進む。

【0068】ステップST29において、抑止キュー28は、抑止していたデータを配送キュー29を介してアプリケーション21に供給し、ステップST26に戻る。

【0069】一方、ステップST27で因果関係がないと判定したときのステップST30において、抑止キュー28に抑止されている次の要素、すなわち次のデータ

10

20

30

40

50

を処理の対象として、ステップST26に戻る。これにより、抑止キュー28に抑止されているデータを次々に処理する。

【0070】以上のように、メンバ2は、他のメンバ2から送信されるデータについて、ベクタタイムスタンプを検査して因果順序を判断することによって、正確な順序でデータを受信することができる。なお、メンバ2が鍵サーバ3から共有鍵を受信した場合も同様である。さらに、鍵サーバ3がメンバ2からデータを受信する場合も同様の処理を実行する。

【0071】また、メンバ2は、データを他のメンバ2に送信するときは、図7に示すステップST31以下の処理を実行する。

【0072】ステップST31において、アプリケーション21は送信すべきデータの暗号化処理を行い、受信モジュール24はこれを受け取って、ステップST32に進む。

【0073】ステップST32において、受信モジュール24は暗号化済みのデータを遅延キュー25に供給して、ステップST33に進む。

【0074】ステップST33において、遅延キュー25にデータが供給されてから出力するまでの時間tの間に共有鍵が配布されたかを判定し、共有鍵が配布されたと判定したときはステップST34に進み、共有鍵が配布されていないと判定したときはステップST36に進む。

【0075】ステップST34において、メンバ2は、保有している共有鍵を新たに配布された共有鍵に更新して、ステップST35に進む。ここで、配布された新しい共有鍵は、古い共有鍵によって暗号化されている。したがって、アプリケーション21が古い鍵を用いて新しい鍵を復号することで、新たな鍵の更新を行う。

【0076】ステップST35において、遅延キュー25に記憶されているデータの送信を中止すると共に、アプリケーション21は新たな共有鍵を用いて送信の対象となっていたデータを再び暗号化し直して、ステップST31に戻る。

【0077】一方、ステップST33で時間tの間に共有鍵が配布されていないと判定したときのステップST36において、遅延キュー25に記憶されているデータを他のメンバ2や鍵サーバ3に送信し、送信処理を終了する。

【0078】このようなステップST31からステップST36の処理を行うことによって、図11で説明した問題を解決することができる。

【0079】例えば図8に示すように、鍵サーバ(Key Server)、メンバA(Member A)、メンバB(Member B)からなるグループがあるものとする。鍵サーバは、新しい共有鍵 K_{new} を生成すると、古い共有鍵 K_{old} を用いて当該新しい共有鍵 K_{new} を暗号化し ($\{K_{new}\}$

K_{old})、これをメンバA及びメンバBに配布する。

【0080】一方、メンバBのアプリケーション21は、古い共有鍵 K_{old} を用いてメッセージMを暗号化し、これを遅延キュー25に供給する。遅延キュー25は、時刻t1において、アプリケーション21から暗号化されたメッセージMを受け取って保持し、時間tのバッファリングを開始する。

【0081】時刻t2において、メンバBは、鍵サーバが配布した $\{K_{new}\}K_{old}$ を受信すると、遅延キュー25に保持されていたデータの送信を中止する。このとき、アプリケーション21は、 $\{K_{new}\}K_{old}$ を復号して、新しい共有鍵 K_{new} を用いてメッセージMを再暗号化する ($\{M\}K_{new}$)。遅延キュー25は、 $\{M\}K_{new}$ を保持すると、再び時間tのバッファリングを開始する。

【0082】時刻t3において、メンバBは、時間tのバッファリングが終了すると、 $\{M\}K_{new}$ を鍵サーバ及びメンバAに送信する。

【0083】これにより、メンバAは、鍵サーバからの新たな鍵 $\{K_{new}\}K_{old}$ を受け取ってから、メンバBからの新たな共有鍵によって暗号化されたデータ $\{M\}K_{new}$ を受信するので、再送要求をすることなくメンバBからのデータを復号することができる。

【0084】すなわち、上記ネットワークシステム1は、このような場合にメンバAがメンバBに対して再送要求を行うことがなくなるので、グループ内のデータの秘密性を保持しつつ、常にリアルタイムでマルチキャスト通信を行うことができる。

【0085】また、上記ネットワークシステム1は、さらにデータの再送要求機構を備えるようにしてもよい。通常、各メンバ2は再送要求に備えて送信済みのデータを一時保存しておく必要があり、従来は、どのくらい保存しておけば良いか不明であり、場合によっては無限に保存しておく必要があった。

【0086】しかし、各メンバ2は、最も遠い受信者たるメンバ2又は鍵サーバ3までの転送時間Tを算出して転送時間Tの間だけデータを保存しておき、再送要求があったときには再送要求先に上記データを再送すればよい。これにより、各メンバ2が無用に多くのデータを保存する負担を減らすことができる。また、各メンバ2は、再送要求の代わりに、復号できなかったデータの有無をアプリケーションに通知するようにしてもよい。

【0087】さらに、上記ネットワークシステム1は、状況に応じて、上述した送信データのバッファリング機構と、再送要求機構とのいずれかを選択して、処理を実行するようにしてもよい。これにより、各メンバ2の使用状況やシステムの処理能力に応じたマルチキャスト通信を行うことができる。

【0088】なお、本発明は、リアルタイムで多数の人が対戦することができるオンラインゲーム、チャット等にも適用することができる。また、高性能な動画を転送

するビデオ会議システム、マルチユーザが参加する仮想空間分散ゲーム等、リアルタイムマルチキャスト通信が使用される場合に適用することができるのは勿論である。

【0089】

【発明の効果】以上詳細に説明したように、本発明に係るネットワークシステム及びデータ送受信方法によれば、各クライアントは、送信対象となる暗号化済みのデータに対して、保証遅延時間に相当する時間の遅延処理を施し、上記遅延処理中に鍵サーバから新たな鍵が配布されたときは、新たな鍵によってデータを再暗号化して当該暗号化されたデータを再び遅延処理の対象とし、上記遅延処理中に鍵サーバから新たな鍵が配布されなかったときは、遅延処理の施された暗号化済みのデータを他のクライアント及び鍵サーバに対して送信することにより、共有鍵の配布や共有鍵を用いたデータの送受信の順番に関係なく、データの秘密性を保持しつつ、リアルタイム性を維持してマルチキャスト通信を行うことができる。

【図面の簡単な説明】

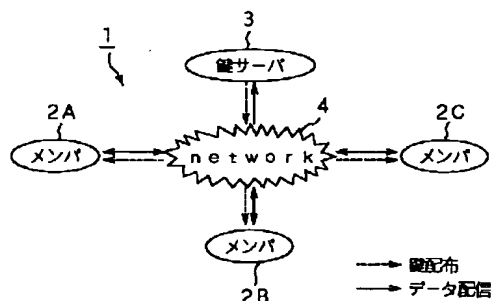
【図1】本発明を適用したネットワークシステムの概略的な構成を示す図である。

【図2】上記ネットワークシステムの具体的なシステム構成を示すブロック図である。

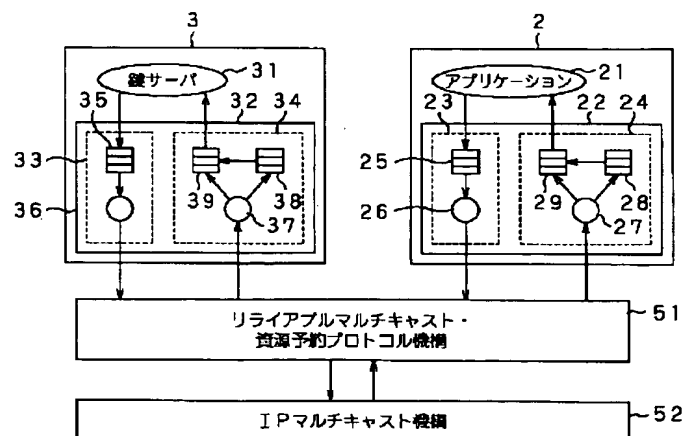
【図3】因果順序による順序付けを説明するための一例を示した図である。

【図4】上記ネットワークシステムによって構成されるグループにあるユーザがメンバとして参加するときの動作を示すフローチャートである。

【図1】



【図2】



【図5】メンバが上記グループから脱退するときの動作を示すフローチャートである。

【図6】メンバが他のメンバから暗号化されたデータを受信するときの動作を示すフローチャートである。

【図7】メンバが他のメンバに対してデータを暗号化して送信するときの動作を示すフローチャートである。

【図8】鍵サーバ、メンバA、メンバBからなるグループにおいて、メンバBがバッファリングしてデータを送信することを説明するための図である。

10 【図9】鍵サーバ、メンバA、メンバBからなるグループにおいて、メンバAが、因果順序による順序付けをしてデータを受信することを説明する図である。

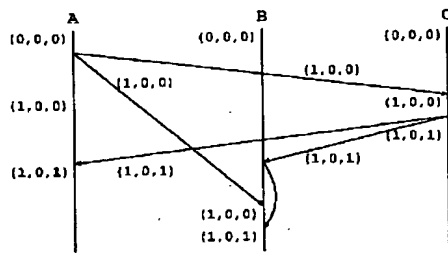
【図10】上記グループにおいて、メンバAが、因果順序による順序付けをしてデータを受信することを説明する図である。

【図11】上記グループにおいて、メンバAが、データを受信する際に、因果順序による順序付けを行うことができない場合を説明する図である。

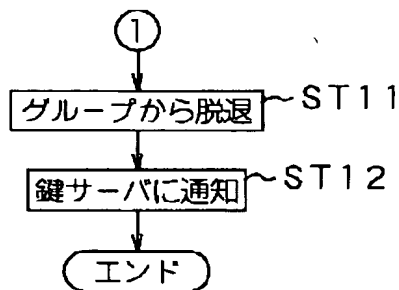
【符号の説明】

20 1 ネットワークシステム、2 メンバ、3 鍵サーバ、21 アプリケーション、22、32 因果順序マルチキャスト機構、23、33 送信モジュール、24、34 受信モジュール、25、35 遅延キュー、26、36 ベクタタイムスタンプ機構、27、37 ベクタタイムスタンプ機構、28、38 抑止キュー、29、39 配送キュー、51 リラヤブルマルチキャスト・資源予約プロトコル機構、52 IPマルチキャスト機構

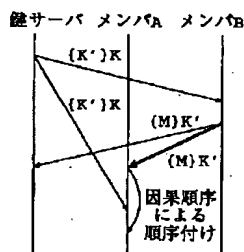
【図3】



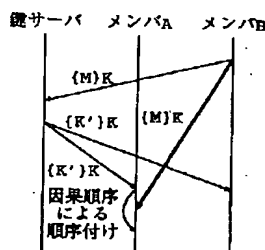
【図5】



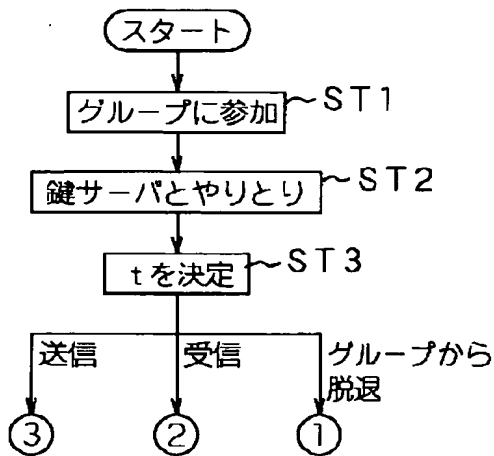
【図9】



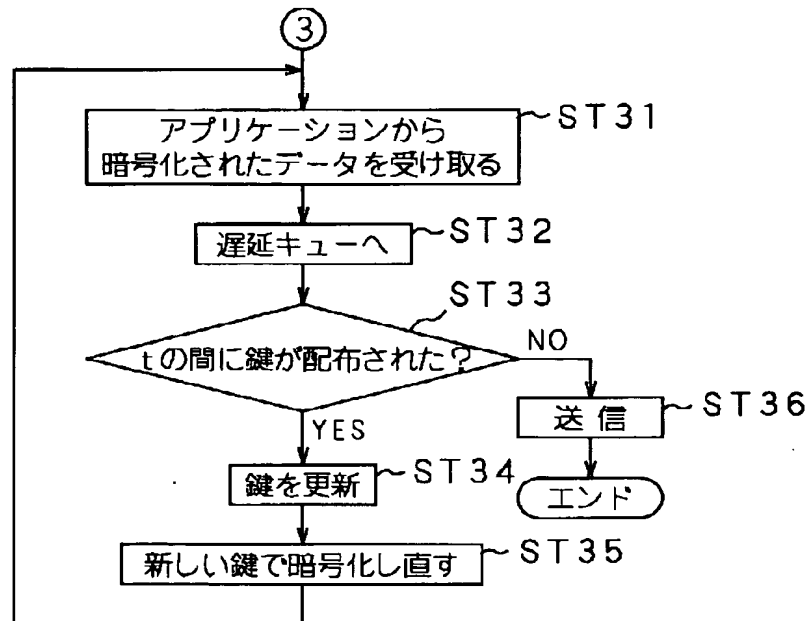
【図10】



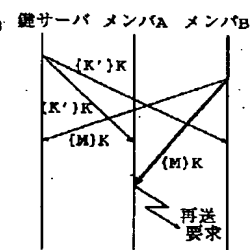
【図4】



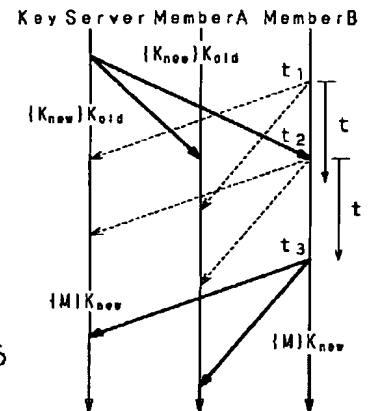
【図7】



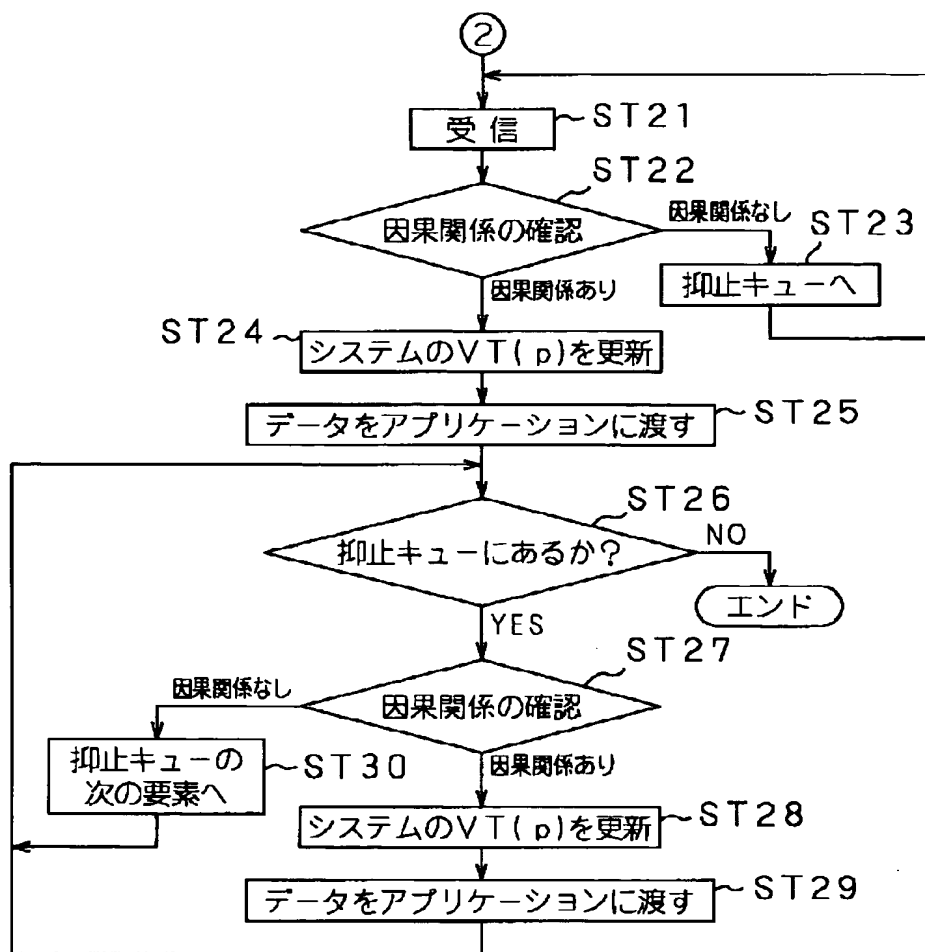
【図11】



【図8】



【図 6】



フロントページの続き

Fターム(参考) 5J104 AA01 AA16 AA34 EA01 EA16
 MA05 NA02 PA07
 5K030 GA15 HA08 HB16 HB18 HC01
 JT02 JT06 LA07 LD06 LD19
 9A001 CC03 EE03 JJ13 JJ18 KK60
 LL03